COMP9414/9814/3411: Artificial Intelligence

16. Course Review

Assessment

Assessable components of the course:

Assignment 1	10%
Assignment 2	8%
Assignment 3	12%
Written Exam	70%

- Exam Template is available on the course Web page
- Exam Questions will be similar in style to the Tutorial Questions

Topics Covered

- Prolog Programming
- Environment/Agent Types
- Search Strategies
- Informed Search (A^*)
- Game Playing
- Logical Agents
- First Order Logic
- Constraint Satisfaction Problems
- Learning and Decision Trees
- Perceptrons
- Neural Networks
- Reasoning under Uncertainty

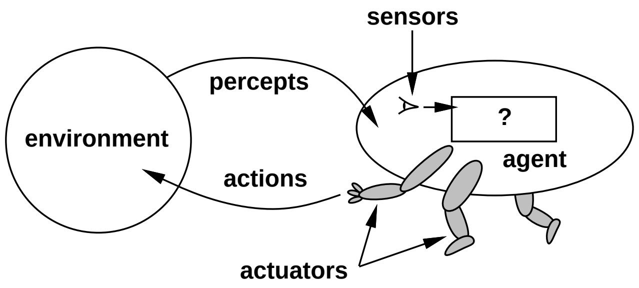
Additional Thursday Topics:

- Reactive Agents
- Evolutionary Computation
- Reinforcement Learning

Not Examinable

- General Game Playing
- Learning Games
- Motion Planning
- Variations on Backprop
- Deep Learning

Agent Model

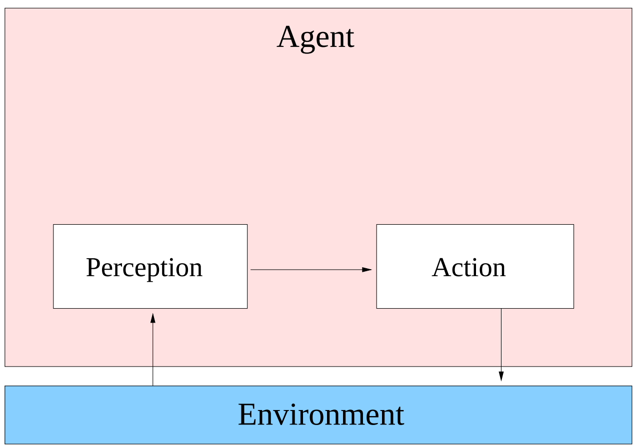


Environment types

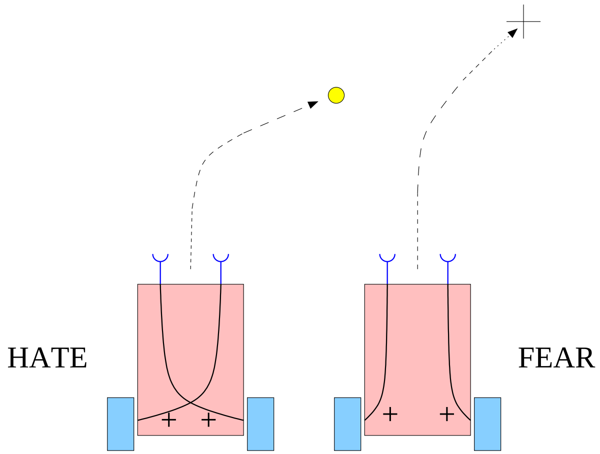
We can classify environments as:

- Fully Observable vs. Partially Observable
- Deterministic vs. Stochastic
- Single-Agent vs. Multi-Agent
- Episodic vs. Sequential
- Static vs. Dynamic
- Discrete vs. Continuous
- Known vs. Unknown
- Simulated vs. Situated or Embodied

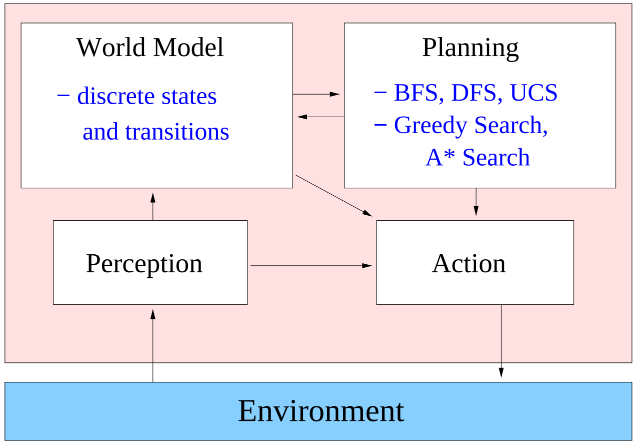
Reactive Agents



Braitenberg Vehicles (Thu only)



Path Search Agent



Path Search Algorithms

General Search algorithm:

- add initial state to queue
- repeat:
  - take node from front of queue
  - test if it is a goal state; if so, terminate
  - "expand" it, i.e. generate successor nodes and add them to the queue

Search strategies are distinguished by the order in which new nodes are added to the queue of nodes awaiting expansion.

Search Strategies

- BFS and DFS treat all new nodes the same way:
  - BFS add all new nodes to the back of the queue
  - DFS add all new nodes to the front of the queue
- (Seemingly) Best First Search uses an evaluation function $f()$ to order the nodes in the queue; we have seen one example of this:
  - UCS $f(n)$ = cost $g(n)$ of path from root to node n
- Informed or Heuristic search strategies incorporate into $f()$ an estimate of distance to goal
  - Greedy Search $f(n)$ = estimate $h(n)$ of cost from node n to goal
  - A^* Search $f(n) = g(n) + h(n)$

Complexity Results for Uninformed Search

Criterion	Breadth-First	Uniform-Cost	Depth-First	Depth-Limited	Iterative Deepening
Time	$O(b^{(d+1)})$	$O(b^{\lceil C^*/\varepsilon \rceil})$	$O(b^m)$	$O(b^l)$	$O(b^d)$
Space	$O(b^{(d+1)})$	$O(b^{\lceil C^*/\varepsilon \rceil})$	$O(bm)$	$O(bl)$	$O(bd)$
Complete?	Yes^1	Yes^2	No	No	Yes^1
Optimal ?	Yes^3	Yes	No	No	Yes^3

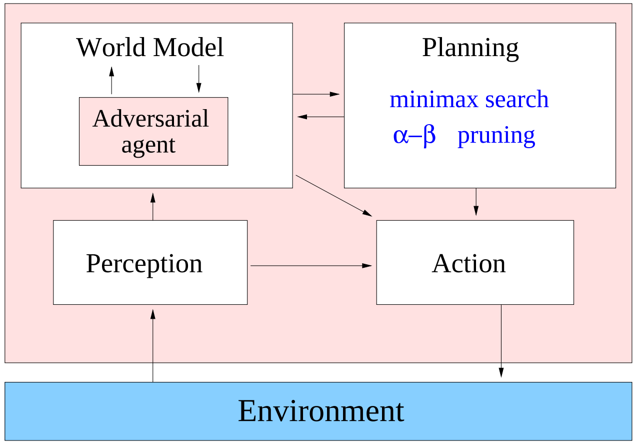
b = branching factor, d = depth of the shallowest solution,
m = maximum depth of the search tree, l = depth limit.

1 = complete if b is finite.

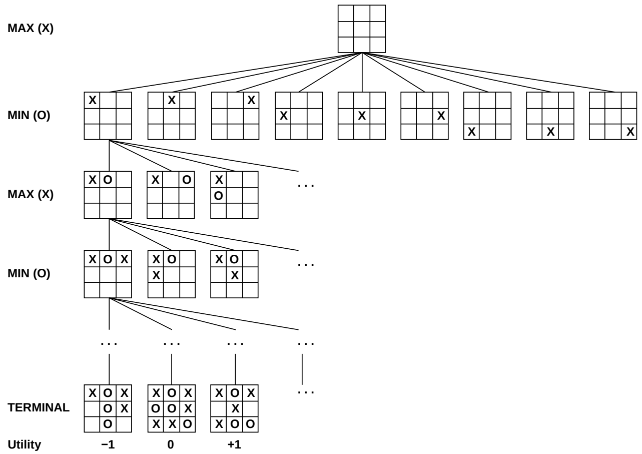
2 = complete if b is finite and step costs $\geq \varepsilon$ with $\varepsilon > 0$.

3 = optimal if actions all have the same cost.

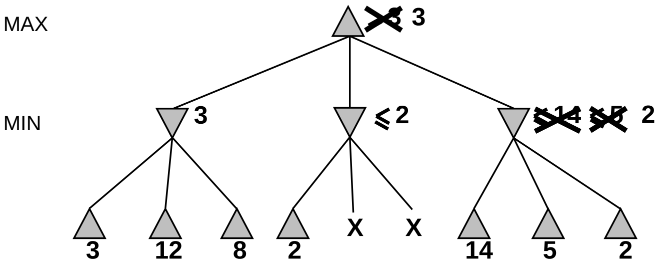
Game Search Agent



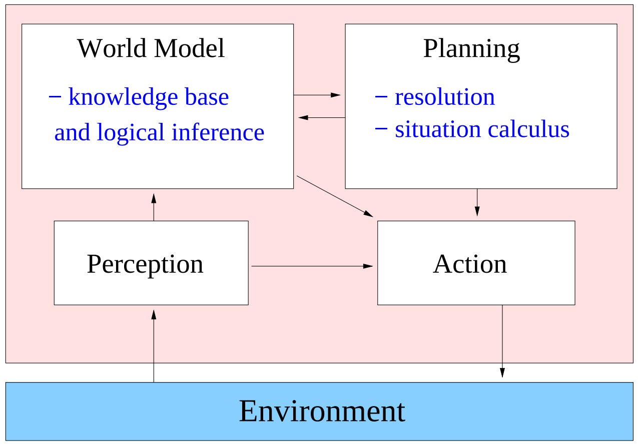
Minimax Search



α-β pruning



Logical Agent



Propositional Logic

A sentence is valid if it is true in all models,
e.g. TRUE, $A \lor \neg A$, $A \Rightarrow A$, $(A \land (A \Rightarrow B)) \Rightarrow B$

Validity is connected to inference via the Deduction Theorem:
$KB \models \alpha$ if and only if $(KB \Rightarrow \alpha)$ is valid

A sentence is satisfiable if it is true in some model
e.g. $A \lor B$, C

A sentence is unsatisfiable if it is true in no models
e.g. $A \land \neg A$

Satisfiability is connected to inference via the following:
$KB \models \alpha$ if and only if $(KB \land \neg\alpha)$ is unsatisfiable
i.e. prove α by *reductio ad absurdum*

Truth Tables

P	Q	¬ P	P ∧ Q	P ∨ Q	P ⇒ Q
F	F	T	F	F	T
F	T	T	F	T	T
T	F	F	F	T	F
T	T	F	T	T	T

Resolution

Conjunctive Normal Form (CNF – universal)

$\text{conjunction of } \underbrace{\text{disjunctions of literals}}_{\text{clauses}}$

e.g. $(A \lor \neg B) \land (B \lor \neg C \lor \neg D)$

Resolution inference rule (for CNF): complete for propositional logic

$$\frac{\ell_1 \lor \cdots \lor \ell_k, \qquad m_1 \lor \cdots \lor m_n}{\ell_1 \lor \cdots \lor \ell_{i-1} \lor \ell_{i+1} \lor \cdots \lor \ell_k \lor m_1 \lor \cdots \lor m_{j-1} \lor m_{j+1} \lor \cdots \lor m_n}$$

where ℓ_i and m_j are complementary literals. e.g.

$$\frac{P_{1,3} \lor P_{2,2}, \qquad \neg P_{2,2}}{P_{1,3}}$$

Resolution is sound and complete for propositional logic.

First Order Logic

Constants	$Gold, Wumpus, [1,2], [3,1]$, etc.
Predicates	$Adjacent(), Smell(), Breeze(), At()$
Functions	$Result()$
Variables	$x,\ y,\ a,\ t, \ldots$
Connectives	$\land \ \lor \ \neg \ \Rightarrow \ \Leftrightarrow$
Equality	$=$
Quantifiers	$\forall \ \exists$

Sentences

Brothers are siblings

$\forall x,y\ Brother(x,y) \Rightarrow Sibling(x,y)$

“Sibling” is symmetric

$\forall x,y\ Sibling(x,y) \Leftrightarrow Sibling(y,x)$

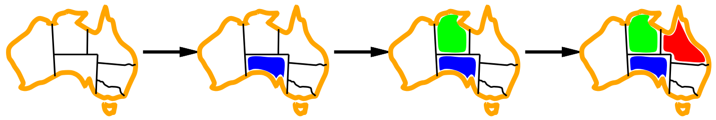
One’s mother is one’s female parent

$\forall x,y\ Mother(x,y) \Leftrightarrow (Female(x) \wedge Parent(x,y))$

A first cousin is a child of a parent’s sibling

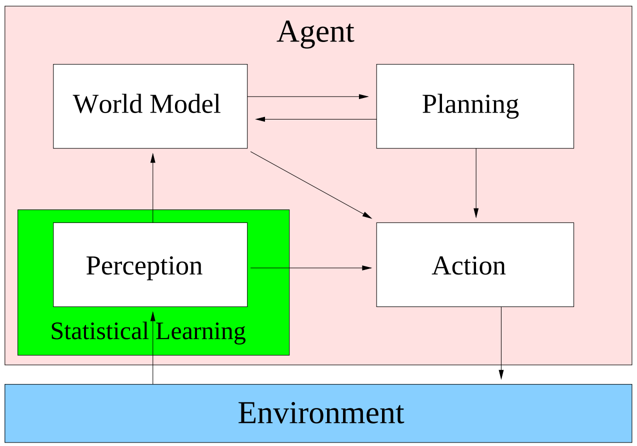
$\forall x,y FirstCousin(x,y) \Leftrightarrow \exists p, ps\ Parent(p,x) \wedge Sibling(ps,p) \wedge Parent(ps,y)$

Constraint Satisfaction Problems



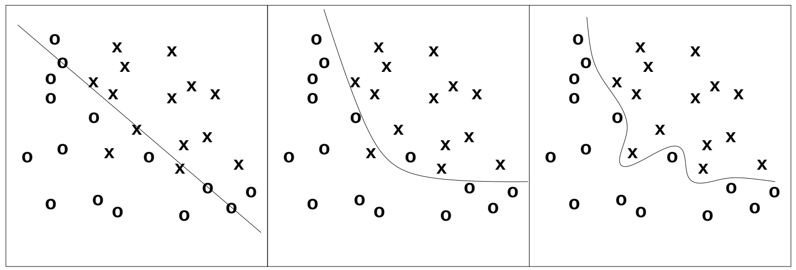
- backtracking search
- enhancements to backtracking search
- local search
  - hill climbing
  - simulated annealing

Statistical Learning Agent



Ockham’s Razor

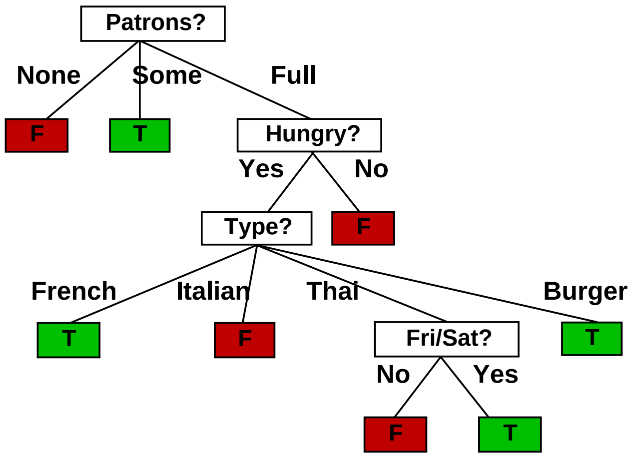
“The most likely hypothesis is the simplest one consistent with the data.”



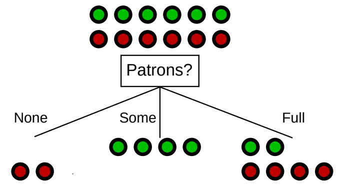
inadequate | good compromise | over-fitting

Since there can be noise in the measurements, in practice need to make a tradeoff between simplicity of the hypothesis and how well it fits the data.

Decision Tree



Choosing an Attribute



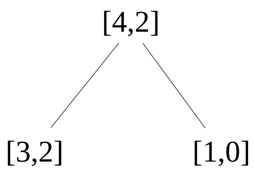
Patrons is a “more informative” attribute than Type, because it splits the examples more nearly into sets that are “all positive” or “all negative”.

This notion of “informativeness” can be quantified using the mathematical concept of “entropy”.

A parsimonious tree can be built by minimizing the entropy at each step.

Minimal Error Pruning

Should the children of this node be pruned or not?



Left child has class frequencies [2,4]

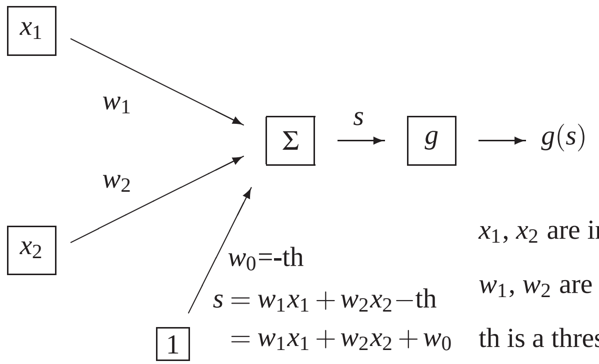
$$E = 1 - \frac{n+1}{N+k} = 1 - \frac{4+1}{6+2} = 0.375$$

Right child has $E = 0.333$
Parent node has $E = 0.444$
Average for Left and Right child is

$$E = \frac{6}{7}(0.375) + \frac{1}{6}(0.333) = 0.413$$

Since $0.413 > 0.375$, children should be pruned.

Rosenblatt Perceptron



w_0=-th

$$s = w_1x_1 + w_2x_2 - \text{th}$$
$$= w_1x_1 + w_2x_2 + w_0$$

x_1, x_2 are inputs

w_1, w_2 are synaptic weights

th is a threshold

w_0 is a **bias** weight

g is transfer function

Perceptron Learning Rule

Adjust the weights as each input is presented.

recall: $s = w_1 x_1 + w_2 x_2 + w_0$

if $g(s) = 0$ but should be 1,

$$\begin{aligned} w_k &\leftarrow w_k + \eta x_k \\ w_0 &\leftarrow w_0 + \eta \end{aligned}$$

$$\text{so} \quad s \leftarrow s + \eta\,(1 + \sum_k x_k^2)$$

if $g(s) = 1$ but should be 0,

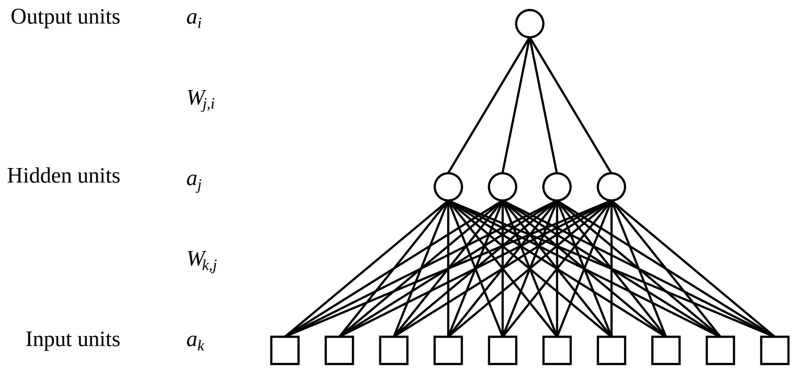
$$\begin{aligned} w_k &\leftarrow w_k - \eta x_k \\ w_0 &\leftarrow w_0 - \eta \end{aligned}$$

$$\text{so} \quad s \leftarrow s - \eta\,(1 + \sum_k x_k^2)$$

otherwise, weights are unchanged. ($\eta > 0$ is called the **learning rate**)

Theorem: This will eventually learn to classify the data correctly, as long as they are **linearly separable**.

Multi-Layer Neural Networks



Gradient Descent

We define an **error function** E to be (half) the sum over all input patterns of the square of the difference between actual output and desired output

$$E = \frac{1}{2} \sum (z - t)^2$$

If we think of E as height, it defines an error **landscape** on the weight space. The aim is to find a set of weights for which E is very low. This is done by moving in the steepest downhill direction.

$$w \leftarrow w - \eta\,\frac{\partial E}{\partial w}$$

Parameter η is called the learning rate.

Probability and Uncertainty

Start with the joint distribution:

	toothache		*¬ toothache*	
	catch	*¬ catch*	*catch*	*¬ catch*
cavity	**.108**	**.012**	**.072**	**.008**
¬ cavity	**.016**	**.064**	**.144**	**.576**

Can compute conditional probabilities:

$$\begin{aligned} P(\neg cavity | toothache) &= \frac{P(\neg\texttt{cavity} \land \texttt{toothache})}{P(\texttt{toothache})} \\ &= \frac{0.016 + 0.064}{0.108 + 0.012 + 0.016 + 0.064} = 0.4 \end{aligned}$$

Bayes' Rule

Product rule $P(a \wedge b) = P(a|b)P(b) = P(b|a)P(a)$

$$\rightarrow \text{Bayes' rule } P(a|b) = \frac{P(b|a)P(a)}{P(b)}$$

Useful for assessing diagnostic probability from causal probability:

$$P(\text{Cause}|\text{Effect}) = \frac{P(\text{Effect}|\text{Cause})P(\text{Cause})}{P(\text{Effect})}$$

e.g., let M be meningitis, S be stiff neck:

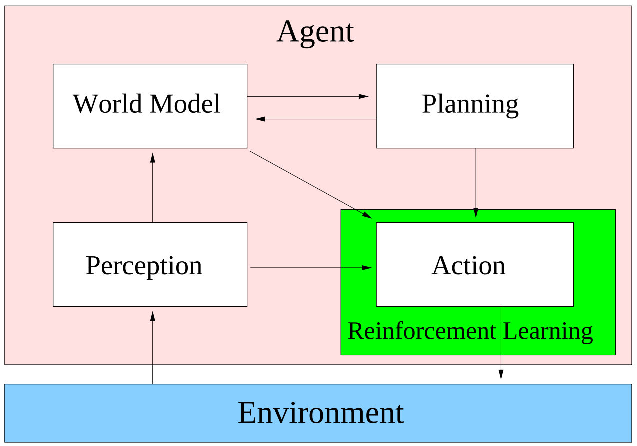
$$P(m|s) = \frac{P(s|m)P(m)}{P(s)} = \frac{0.8 \times 0.0001}{0.1} = 0.0008$$

Note: posterior probability of meningitis still very small!

Evolutionary Computation (Thu only)

- use principles of natural selection to evolve a computational mechanism which performs well at a specified task.
- start with randomly initialized population
- repeated cycles of:
  - evaluation
  - selection
  - reproduction + mutation
- any computational paradigm can be used, with appropriately defined reproduction and mutation operators

Reinforcement Learning Agent



Q-Learning (Thu only)

For each $s \in S$, let $V^*(s)$ be the maximum discounted reward obtainable from s, and let $Q(s,a)$ be the discounted reward available by first doing action a and then acting optimally.

Then the optimal policy is

$$\pi^*(s) = \text{argmax}_a Q(s,a)$$

where $$Q(s,a) = r(s,a) + \gamma V^*(\delta(s,a))$$

then $$V^*(s) = \max_a Q(s,a),$$

so $$Q(s,a) = r(s,a) + \gamma \max_b Q(\delta(s,a), b)$$

which allows us to iteratively approximate Q by

$$\hat{Q}(s,a) \leftarrow r + \gamma \max_b \hat{Q}(\delta(s,a), b)$$

Beyond COMP9414/9814/3411

- COMP9444 Neural Networks and Deep Learning
- COMP9417 Machine Learning and Data Mining
- COMP4418 Knowledge Representation and Reasoning
- COMP3431 Robotic Software Architecture
- COMP9517 Machine Vision
- 4th Year Thesis topics

Possible 4th Year Projects

- Image Processing
  - generation of images by Deep Convolutional Neural Networks
- Language Processing
  - autoencoders, word2vec, LSTM
  - linguistic analysis
- Evolutionary Automatic Programing, with HERCL
  - combination of Linear Genetic Programming with stack-based GP
  - multiple tasks, transfer learning, evolving modularity
- combining Evolutionary Computation with Deep Learning
  - evolving memristor or spiking networks
  - evolutionary art, co-evolving images or text

COMP9414/3411 Artificial Intelligence

QUESTIONS?

COMP9414/3411 Artificial Intelligence

GOOD LUCK!